Report for assignment1

U6532004

Statement

There is no optimal solution for this problem without central control, we can only approach the optimal solution as much as possible.
I used the spherical model, which was referenced to the Examples video for this assignment. When COMMUNICATING with my friends, we conclude this sphere model performs best.

Introduction

The basic structure of the program is an infinite loop, and each loop is divided into four steps below:

1.Detect energy globe
2.Send a message
3.Receiving information
4.Set the destination and throttle

This report is an illustration of assignment 1 for COMP2310, including with test documentation also. The whole progress of vehicles is run by consensus algorithm to pass massage through individuals, every vehicle will refresh their information time by time when in the main task loop and keep the latest message as receiving message. Furthermore, I will introduce hypothesis which doesn't apply yet, but I have a frame and concept with.

Core Design concept

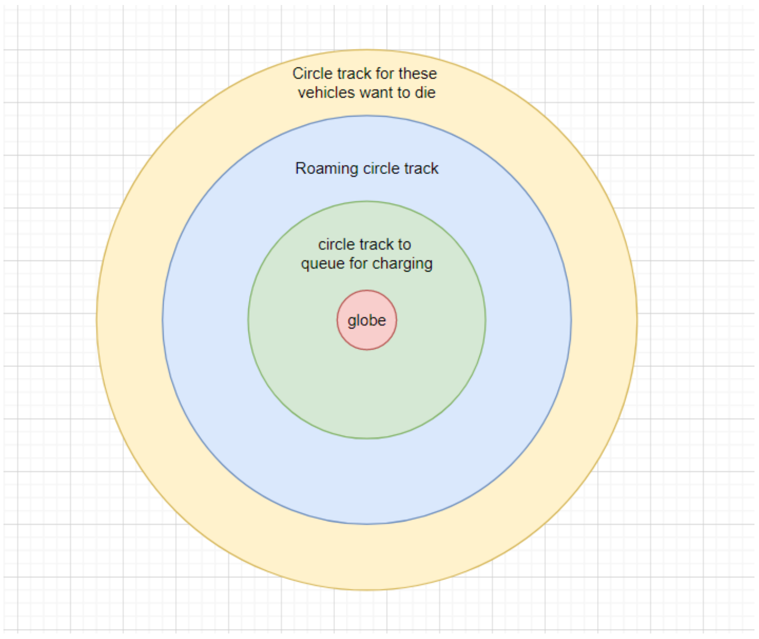
Actually, the problem of this project is that there is no central control system, which means that each car is completely equal and independent. So, the information should spread one by one when any globes found as soon as possible. They know where the globe is and they can decide whether they should go for charging.

1.Multiple Energy Globes Charging Decision

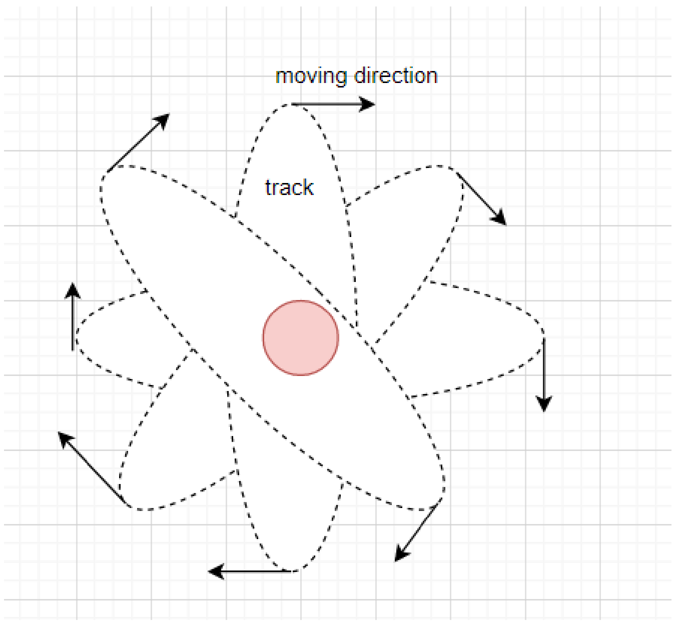
My strategy to find these globes is very simple that if any vehicle found globe around them then send message out to the others. Then all of vehicles can know where the globe is. But the problem is from stage 3, random multiple energy globes will appear and disappear randomly and amount of globes is not absolute. In this situation, my current decision is to randomly assign the coordinates of an energy ball as the destination at the beginning and keep looking for the near energy ball on the way and change destination if find closer globe. This strategy allows vehicles to look for a closer energy ball to charge on the way to charge, making the entire transportation network more flexible.

1.1 Circle Track Meaning

All vehicles move around globe as a sphere. When energy of a vehicle is below 0.6 (the most stable number after testing, introduced in 6th part), vehicle will detect whether there are more than 4 vehicles want to charge, if yes, then all vehicles who want to charge will go straight to globe. This step is to increase charging efficiency. Because if they are few than 4 vehicles want to charge, they will go to the green circle track(shown in below graph) and roaming until they have order to charge. In this situation, the vehicles who want to charge can be the nearest group to the globe. For these vehicles with much energy, they will stay in blue circle track and charge when they needed. Yellow circle is a dying roaming circle for those vehicles decide to disappear. The reason why keeps them as a circle rather than assign them out of globe is aesthetic and form unity, circle track is beautiful at all.


Circle track for these vehicles want to die
Roaming circle track
circle track to queue for charging
globe

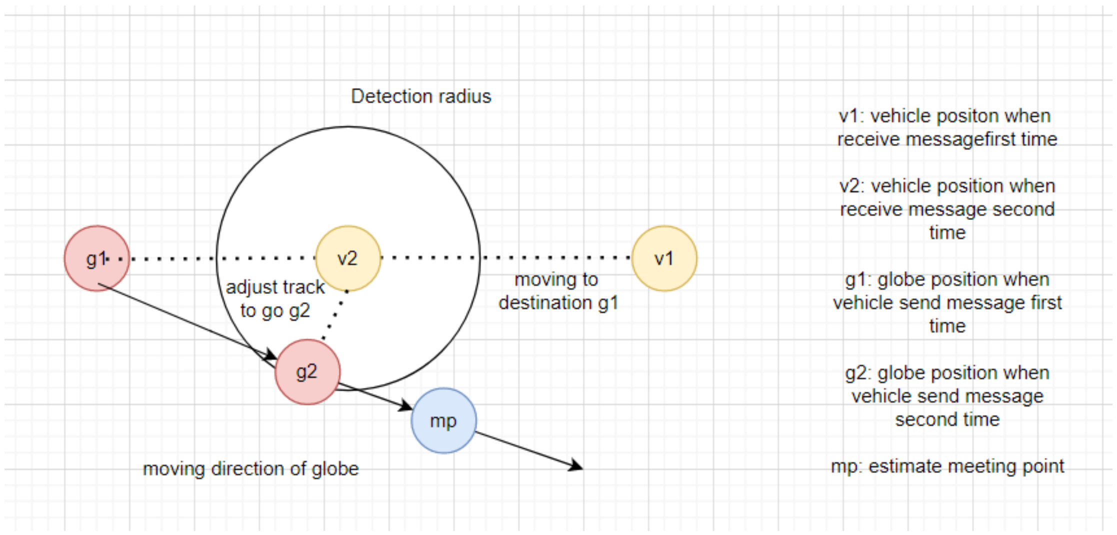
1.2 Circle Track For Individuals



Each vehicle will calculate their own angle through plus Pi to do circle movement. The dynamic rounding by globe makes charging more stable as they are always in movement and can react immediately when they need to charge. Otherwise, there is a time waste as they accelerate from rest and raise their acceleration to the target threshold. There turning around angle was divided by number of vehicles which means they will go around one circle after all vehicles called and loop again which make sure their turning angle is same. (Left graph)

2.Vehicles Moving Motivation When Detected

In this project, real-time updates of vehicle destinations are very important. No matter the vehicle is roaming around circles or going to charge, globe itself will move as well. It is possible to meet energy globe in the first-time scheduled position. My solution for that is call a detection program of globe when any vehicle task called. Vehicle will detect globe even they have been set destination and change destination immediately if they found any closer globes. Like the graph below, v1 go to g1 at the beginning. Then when it moves to v2 position, it detects g2 which is actually g1 moving here. Vehicle will adjust its track and finally meet with globe in mp.


Detection radius
g1
v2
v1
adjust track to go g2
moving to destination g1
g2
mp
moving direction of globe
v1: vehicle positon when receive messagefirst time
v2: vehicle position when receive message second time
g1: globe position when vehicle send message first time
g2: globe position when vehicle send message second time
mp: estimate meeting point

3.Message Type and Time

There are two kinds of message, one is local message another is replaced message. The initial message is local message, each detection will send local message out. Each vehicle has its local time when established, when they receive any messages from the others, they will compare the time of message time and their own time. If the time of message is newer, they will assign replaced message with content of local message. And do operations of replaced message to ensure originality of local message. The replaced message will save and send until they find received message is order than them. This strategy aims to protect local message transfer local message out when it is latest.

Sometimes, vehicle find new energy globes in the message they get from others. This situation is actually very complicated, because when two vehicles are talking about the same energy ball, but not at the same time, and it is difficult to tell whether this globe is the same energy globe in the message.

We can use the time stamp to determine which energy globe information is latest. However, this update cannot be based on the timestamp as the only criteria because there is a situation where vehicle A has a message including a globe position but there is a long time since the discovery. And there is a vehicle B get close to vehicle A , the two cars will exchange information. Whose information should choose here? Answer is that they can compare their clock to find whose massage is latest.
There is no absolute concurrency at the world, so do vehicle tasks. Their time clock is not created at the same time, which can cause trouble in message transmitting. When a vehicle receives older message, they will put their clock on it to refresh time stamp. After several times of transmitting, all clock of vehicles should be unified.

------ Error Discussion for section 3-Message type and time
However, this unifying has stability to ensure the situation won't go wrong after several minutes as all of them are unified. But this unifying needs all vehicles contact with the first vehicle at least once, so it can cause error in the beginning stage.
Sometimes there will be a vehicle out of control and doesn't get charge when it has been assigning alive attribute. Such a runaway car would monopolize one quota of surviving amount and disappear eventually, resulting in one a smaller number of remaining survivors.

4.Survive decision

In stage 4, vehicles need to decide who can survive and who needs to die for achieve target element number. The solution of mine to this problem is to simply assign surviving numbers from the first vehicle until the number of vehicles assigned to alive run out to 0. All of the other vehicle who didn't assign to survive should go the most far circle track and wait to die without charging.

This strategy needs to assign surviving attribute in the beginning of creating vehicle task. Otherwise, vehicles cannot get to know who has died and who still on the way to die because of message passing takes time. So this is why I didn't assign dying message attribute when a vehicle is dying. Their fortunate was decided at the beginning.

5.Traffic problem (Hypothesis)

However, a problem is traffic trouble. For example, if too many vehicles go to one globe, too crowded roads can hinder other vehicles and cause some vehicles to be unable to charge in time and disappear.

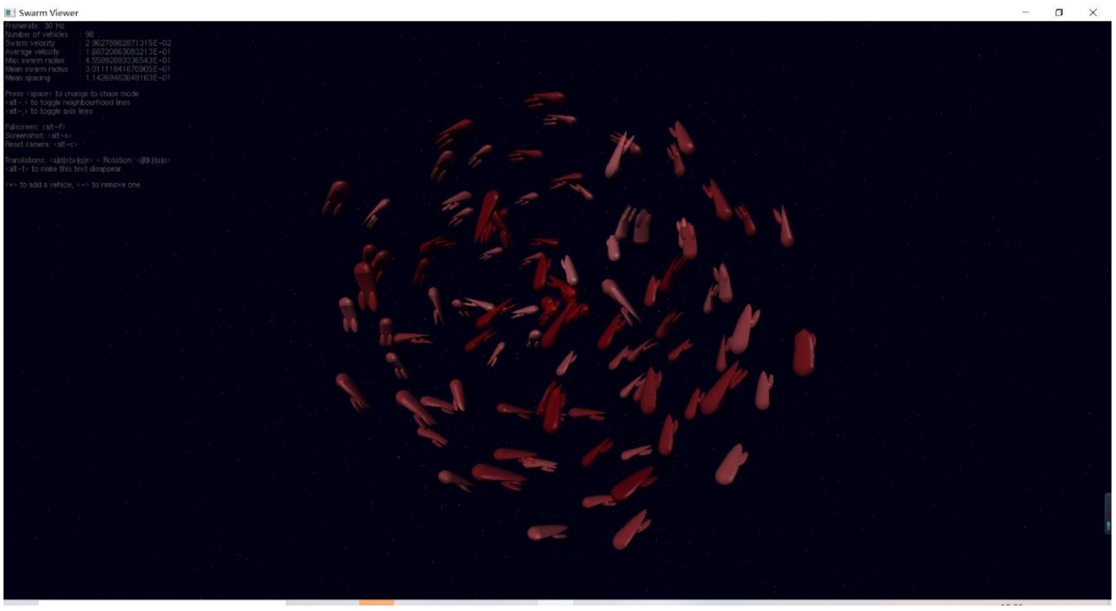
I didn't apply these functions to solve traffic problem in my program, because they are just a hypothesis, the way to implement them are too difficult and after trying I gave up. The number of vehicles can only stay nearly 160 finally.

One is the destination count function, I wish to add an attribute into inter vehicle message to show the destination of these vehicles and a function to calculate the number of those vehicles who want to charge to each globe. Then when assign these vehicles actual destination, they will be set to the globe with least vehicles around. But the reason I cannot implement it is some globes with least vehicles around may also the most far globe from current vehicle, current vehicle may run out of energy on the way to charge. If I want to solve it, there should be a weighted calculation for distance.

Weighted calculation is a comprehensive comparison of the number of vehicles around the destination and distance, from which to choose the more suitable as current vehicle own destination. If a vehicle is below specific energy rate like 0.3, it will carry a message with emergency, all the other vehicles should give way to it.

Solution of mine was mentioned in 1.1 and 1.2, I assign them to randomly nearest globe and make them roam in a circle track. This can cause traffic busy when there are too many vehicles and I think this is why my final surviving elements cannot over than 160.

6.Test and maintenance



```
Framerate:  30 Hz
Number of vehicles      : 96
Swarm velocity          : 2.96276982871315E-02
Average velocity        : 1.68720863083213E-01
Max swarm radius        : 4.55892893336543E-01
Mean swarm radius       : 3.01111841670905E-01
Mean spacing            : 1.14269463649163E-01

Press <space> to change to chase mode
<alt-.> to toggle neighbourhood lines
<alt-,> to toggle axis lines

Fullscreen: <alt-f>
Screenshot: <alt-s>
Reset camera: <alt-c>

Translations: <a|d|s|w|q|e> - Rotation: <j|l|k|i|u|o>
<alt-t> to make this text disappear

<+> to add a vehicle, <-> to remove one
```

Image of testing screenshot

Initial number	Target number	Duration	Test times	Average remaining number
64	40	3 mins	5	40
128	100	3 mins	5	100
160	140	3 mins	5	139.8
190	150	3 mins	5	149.3
230	200	3 mins	5	159.4
258	200	3 mins	5	158.9

Test for size of vehicles

-----------*Analysis*

Sometimes, there will be one or two vehicles disappear at very beginning, they takes the surviving amount and die which will affect the final surviving amount. Because the normal vehicles' surviving amount reduce. According to my observation, all the situation that we ended up with a smaller number of survivors than expected, all because there were cars that took the quota at the beginning and none of them survived as planned.

Charging warning value	Initial number	Duration	Test time	Average Remaining number
0.7	100	3 mins	5	99.4
0.6	100	3 mins	5	100
0.5	100	3 mins	5	97.1

Test for charging warning value of vehicles

-----------Analysis

I test this charging value manually to find which is suitable, it is apparently to find 0.6 works best. Through my understanding, I think 0.7 is too high that vehicles cannot do roaming well as they are always on charging way and cause busy traffic. 0.5 is too low that vehicles may not keep survive when they are on the way to charge.

Reference

Consensus Algorithm https://en.wikipedia.org/wiki/Consensus_decision-making ↵